UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/628,726	07/28/2003	Suresh Marisetty	884.108US2	3999

21186 7590 10/16/2006

SCHWEGMAN, LUNDBERG, WOESSNER & KLUTH, P.A.
P.O. BOX 2938
MINNEAPOLIS, MN 55402

EXAMINER	
MASKULINSKI, MICHAEL C	

ART UNIT	PAPER NUMBER
2113	

DATE MAILED: 10/16/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/628,726
Filing Date: July 28, 2003
Appellant(s): MARISETTY ET AL.

Ann M. McCrackin
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed December 1, 2005 appealing from the Office action mailed September 15, 2005.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings, which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6,308,285 B1	**Bowers**	**10-2001**
6,065,078	**Falik et al.**	**5-2000**

5,892,898 **Fujii et al.** **4-1999**

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 101

Claims 24-26 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. Claim 24 claims a recording medium on which a program is stored and variations thereof. These claims therefore are interpreted as recording a program per se. In order to overcome this rejection, language, specifically stating the claim, **must be** limited to a computer program stored on a computer recordable medium executing on a computer.

Claim Rejections - 35 USC § 102

Claims 5-16 and 24-26 are rejected under 35 U.S.C. 102(b) as being anticipated by Bowers, U.S. Patent 6,308,285 B1.

Referring to claim 5:

a. In column 3, lines 45-48, Bowers discloses that because the processors control the functioning of the system generally under the control of software programming, memory is coupled to the processors to store and to facilitate execution of these programs (non-volatile memory to store an error handling routine and an idle routine). Further, in column 4, lines 57-59, Bowers discloses that an ACPI-compliant operating system interprets the SCI interrupt and

prepares to place the identified processor into a sleep mode (an error handling routine and idle routine).

b. In column 2, lines 60-62, Bowers discloses that after the failed processor is replaced all the processors are awakened and the computer is returned to normal operation without the need to reboot the computer (said error handling routine to permit a computer system to continue operating when an error is detected).

c. In column 5, lines 2-9, Bowers discloses that a controller generates a stop clock signal STPCLK# and a sleep signal SLP#. These signals are delivered to each processor via a respective bus. These signals place the processors into a low power state so that they stop providing internal clock signals to all units except the bus unit and the APIC unit. The processors also stop executing commands and tri-state some outputs (a plurality of slave processors to execute the idle routine). Further, in Figure 1, Bowers discloses that the plurality of slave processors are included in the computer system.

d. In column 5, lines 2-3, Bowers discloses a controller (monarch processor included in the computer system) that generates a stop clock signal STPCLK# and a sleep signal SLP# (error handling routine to correct an error).

Referring to claim 6, in column 4, lines 57-65, Bowers discloses that an ACPI-compliant operating system interprets the SCI interrupt and prepares to place the identified processor into a sleep mode. Specifically, the processor will be placed into an ACPI "S2" sleep state. To achieve this result, the operating system services the SCI

interrupt by calling a _PTS routine stored in the ROM/BIOS (a system abstraction layer located in the non volatile memory wherein the system abstraction layer includes the error handling routine).

Referring to claim 7, in column 7, lines 4-6, Bowers discloses that when all removals or replacements have been made, the controller will begin the reinitialization sequence to return the computer to normal operation (wherein the monarch processor is capable of sending a wake up signal to the plurality of slave processors to exit the rendezvous state).

Referring to claim 8:

a. In Figure 2, Bowers discloses a multiprocessor computer system (a plurality of processors) and a PAL controller (a monarch processor).

b. In column 5, lines 2-9, Bowers discloses that a controller generates a stop clock signal STPCLK# and a sleep signal SLP#. These signals are delivered to each processor via a respective bus. These signals place the processors into a low power state so that they stop providing internal clock signals to all units except the bus unit and the APIC unit. The processors also stop executing commands and tri-state some outputs. Since these signals control the processors and are at the processor level there inherently exists a processor abstraction layer coupled to the plurality of processors.

c. In column 4, lines 57-67 continued in column 5, line 1, Bowers discloses that an ACPI-compliant operating system interprets the SCI interrupt and prepares to place the identified processor into a sleep mode. Specifically, the

processor will be placed into an ACPI "S2" sleep state. To achieve this result, the operating system services the SCI interrupt by calling a _PTS routine stored in the ROM/BIOS (an operating system layer coupled to the system abstraction layer). The routine generates a "start" signal and delivers it to the controller via the bridge/memory controller. A layer between the _PTS routine and the controller so that there is a system abstraction layer coupled to the processor abstraction layer is inherent to the system.

d. In column 4, lines 57-67 continued in column 5, lines 1-9, Bowers discloses a sleep state signal that is passed from the operating system to the processors (an interrupt signaling mechanism coupled to the processor abstraction layer, the system abstraction layer, and the operating system layer to initiate a rendezvous state). Further, in column 7, lines 4-6, Bowers discloses that when all removals or replacements have been made, the controller will begin the reinitialization sequence to return the computer to normal operation (ending the rendezvous state).

e. In column 5, lines 2-9, Bowers discloses placing the processors in a sleep state and having a controller not in the sleep state (said rendezvous state being a state where all but one of said processors in said plurality of processors are idle).

Referring to claim 9, in column 3, lines 45-48, Bowers discloses that because the processors control the functioning of the system generally under the control of software programming, memory is coupled to the processors to store and to facilitate execution of these programs (the processor abstraction layer is located in the non volatile memory

and is executed by the plurality of processors, and the system abstraction layer is located in the non volatile memory and is executed by the plurality of processors). Further, in column 4, lines 40-42, Bowers discloses an operating system of the computer (the operating system layer is located in the system memory and executed by the plurality of processors).

Referring to claim 10, in column 4, lines 57-67 continued in column 5, line 1, Bowers discloses that an ACPI-compliant operating system interprets the SCI interrupt and prepares to place the identified processor into a sleep mode. Specifically, the processor will be placed into an ACPI "S2" sleep state. To achieve this result, the operating system services the SCI interrupt by calling a _PTS routine stored in the ROM/BIOS (error handling routine included in the system abstraction layer). The routine generates a "start" signal and delivers it to the controller (monarch processor) via the bridge/memory controller. Further, in column 5, lines 10-13, Bowers discloses that after the processors have been placed into the sleep state (rendezvous state), the controller delivers a reset signal RESET# to the processor being removed, followed by de-assertion of the PWRGOOD signal (the monarch processor executing an error handling routine included in the system abstraction layer upon initiation of the rendezvous state).

Referring to claim 11, in column 5, lines 2-9, Bowers discloses that a controller generates a stop clock signal STPCLK# and a sleep signal SLP#. These signals are delivered to each processor via a respective bus. These signals place the processors into a low power state so that they stop providing internal clock signals to all units

except the bus unit and the APIC unit. The processors also stop executing commands and tri-state some outputs (the processor abstraction layer includes a functional module for error handling).

Referring to claim 12:

a. In Figure 2, Bowers discloses a multiprocessor computer system (a plurality of processors).

b. In column 3, lines 45-48, Bowers discloses that because the processors control the functioning of the system generally under the control of software programming, memory is coupled to the processors to store and to facilitate execution of these programs (a processor abstraction layer is located in a non volatile memory coupled to the plurality of processors, and a system abstraction layer located in the non volatile memory).

c. Further, in column 4, lines 40-42, Bowers discloses an operating system of the computer (an operating system layer located in a system memory coupled to the plurality of processors).

d. In column 5, lines 2-9, Bowers discloses that a controller generates a stop clock signal STPCLK# and a sleep signal SLP#. These signals are delivered to each processor via a respective bus. These signals place the processors into a low power state so that they stop providing internal clock signals to all units except the bus unit and the APIC unit. The processors also stop executing commands and tri-state some outputs. An interface between the signals

generated by the controller and the processors so that there is a processor abstraction layer coupled to the plurality of processors is inherent to the system.

e. In column 4, lines 57-67 continued in column 5, line 1, Bowers discloses that an ACPI-compliant operating system interprets the SCI interrupt and prepares to place the identified processor into a sleep mode. Specifically, the processor will be placed into an ACPI "S2" sleep state. To achieve this result, the operating system services the SCI interrupt by calling a _PTS routine stored in the ROM/BIOS. The routine generates a "start" signal and delivers it to the controller via the bridge/memory controller. An interface between the _PTS routine and the controller so that there is a system abstraction layer coupled to the processor abstraction layer is inherent to the system.

f. In column 4, lines 57-67 continued in column 5, lines 1-9, Bowers discloses a sleep state signal that is passed from the operating system to the processors (an interrupt signaling mechanism coupled to the processor abstraction layer, the system abstraction layer, and the operating system layer for initiation of a rendezvous state to initiate a rendezvous state). Further, in column 7, lines 4-6, Bowers discloses that when all removals or replacements have been made, the controller will begin the reinitialization sequence to return the computer to normal operation (ending the rendezvous state upon receiving a signal that error handling is completed).

g. In column 5, lines 2-9, Bowers discloses placing the processors in a sleep state and having a controller not in the sleep state (said rendezvous state being a state where all but one of said processors in said plurality of processors are idle).

Referring to claim 13, in column 4, lines 57-67 continued in column 5, lines 1-9, Bowers discloses a sleep state signal that is passed from the operating system to the processors (wherein the signal from the operating system to end the rendezvous state is an interrupt).

Referring to claim 14, In column 4, lines 57-67 continued in column 5, lines 1-9, Bowers discloses a sleep state signal that is passed from the operating system to the processors (the processor abstraction layer is capable of sending a signal to the system abstraction layer to enter the rendezvous state). Further, in column 7, lines 4-6, Bowers discloses that when all removals or replacements have been made, the controller will begin the reinitialization sequence to return the computer to normal operation (performing error handling upon entering the rendezvous state).

Referring to claim 15:

a. In column 7, lines 1-4, Bowers discloses that if multiple processors are to be removed or replaced these processors may be identified by a failure detection system or by the user via a software or hardware interface (detecting an error by one processor included in a multiple processor system).

b. In column 5, lines 2-13, Bowers discloses placing the processors in a sleep state (entering a rendezvous state in which all processors but the one processor included in the multiple processor system are idle).

c. In column 5, lines 10-13, Bowers discloses that after the processors have been placed into the sleep state (rendezvous state), the controller delivers a reset signal RESET# to the processor being removed, followed by de-assertion of the PWRGOOD signal (correcting the error using the one processor).

d. In column 2, lines 60-62, Bowers discloses that after replacement of the processor, all processors are awakened and the computer is returned to normal operation without the need to reboot the computer (resuming normal operation).

Referring to claim 16, in column 5, lines 2-13, Bowers discloses placing the processors in a sleep state (requesting a plurality of processors included in the multiple processor system to enter an idle state) and that after the processors have been placed into the sleep state (waiting until the plurality of processors have entered the idle state), the controller delivers a reset signal RESET# to the processor being removed, followed by de-assertion of the PWRGOOD signal.

Referring to claim 24:

a. In column 5, lines 2-9, Bowers discloses placing the processors in a sleep state and having a controller not in the sleep state (attempting to correct an error by a detecting processor included in a multiple processor system).

b. In column 4, lines 60-62, Bowers discloses that the operating system services the SCI interrupt (error) by calling a _PTS routine stored in the ROM/BIOS (on failure, executing firmware code operatively coupled to all the processors included in the multiple processor system to correct the error).

c. In column 5, lines 2-9, Bowers discloses placing the processors in a sleep state and having a controller not in the sleep state (on failure, entering a rendezvous state to correct the error, said rendezvous state being a state where all but one of said processors included in the multiple processor system are idle).

Referring to claim 25, in column 5, lines 2-9, Bowers discloses placing the processors in a sleep state (wherein all but the one of the processors included in the multiple processor system are executing a spin loop) and having a controller not in the sleep state.

Referring to claim 26, in column 5, lines 2-9, Bowers discloses that a controller generates a stop clock signal STPCLK# and a sleep signal SLP#. These signals are delivered to each processor via a respective bus. These signals place the processors into a low power state so that they stop providing internal clock signals to all units except the bus unit and the APIC unit (informing a processor abstraction layer when all but one of the processors included in the multiple processor system have entered the idle state).

Claims 18 and 19 are rejected under 35 U.S.C. 102(e) as being anticipated by Falik et al., U.S. Patent 6,065,078.

Referring to claim 18:

a. In column 1, lines 34-48, Falik et al. disclose that the debugger interface in combination with the host computer sends a debugger command to at least one of the plurality of processors (attempting to correct an error by a detecting processor included in a multiple processor system).

b. In column 4, lines 41-47, Falik et al. disclose sending an interrupt to a processor, which stops the execution of the application program and starts to execute the monitor (on failure, executing firmware code operatively coupled to all the processors included in the multiple processor system to correct the error).

c. In column 7, lines 26-30, Falik et al. disclose that the interrupt control module issues an ISE interrupt request to either a specific one of the processors or to multiple processors (on failure, entering a rendezvous state to correct the error, said rendezvous state being a state where all but one of the processors included in the multiple processor system are idle).

Referring to claim 19:

a. The debugger interface of Falik et al. is pre-designated making it the monarch processor from the processors included in the multiple processor system.

b. In column 4, lines 41-47, Falik et al. disclose sending an interrupt to a processor, which stops the execution of the application program (signaling slave processors included in the multiple processor system to execute a spin loop) and starts to execute the monitor (correcting the error by the monarch processor).

d. In column 2, lines 43-46, Falik et al. disclose that after the phase of debugging software for the processors of the multiprocessor integrated circuit, the multiprocessor integrated circuit would, in most cases, no longer interact with the host computer (resuming normal operation by the plurality of processors).

Claim Rejections - 35 USC § 103

Claim 17 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bowers, U.S. Patent 6,308,285 B1, and further in view of Fujii et al., U.S. Patent 5,892,898. In column 2, lines 49-62, Bowers discloses removing or replacing one or more of the processors for various reasons. However, Bowers doesn't explicitly disclose determining if the error is a severe error and only upon determining that the error is a severe error, entering the rendezvous state. In column 2, lines 14-19, Fujii et al. disclose an event message that corresponds to an event type that includes at least one event type selected from the group consisting of an information event type, a warning event type, and an error event type. It would have been obvious to one of ordinary skill at the time of the invention to include the event categorizing method of Fujii et al. into the system of Bowers. A person of ordinary skill in the art would have been motivated to make the modification because an error event type is used to report a non-recoverable problem (see Fujii et al.: col.2, lines 62-63) and a warning event type is used to indicate some kind of recoverable anomaly (see Fujii et al.: col. 2, lines 60-62). Knowing the severity determines what action should be taken (removal of a processor) (see Fujii et al.: col. 2, lines 48-55).

(10) Response to Argument

On page 13, under section *7.3*, the Appellant argues, "These claims are therefore directed to an article of manufacture, including a system 500 executing code stored in a system memory 540 (see page 14 of the Application, lines 15-16), and as such, clearly fall into one of the four acceptable statutory categories of patentable subject matter."

The Examiner respectfully disagrees. A system and executable code may be an article of manufacture and a machine, but it is not necessarily true vice versa. A machine is not always a computer and a machine-accessible medium can be a piece of paper being scanned. Further, it is unclear as to whether the data is instructions or code and whether it is even stored on a medium.

On page 14, under section *7.4*, the Appellant argues, "Given the method of operation disclosed in Bowers, the Office Action rejects the instant claims by attempting to characterize Bowers' controller as a 'monarch processor.' However, this is inappropriate. First, because Bowers makes a clear distinction between the 'controller' and the 'processors.' It is only the controller in Bowers, and not the processors, that can access data used to put the processors to sleep." The Examiner respectfully disagrees. In column 4, lines 41-67 continued in column 5, lines 1-9, Bowers discloses an operating system of computer 30 that sends signals to controller 112 to place the processors in a sleep state. The combination of an operating system with instructions and a controller that executes the instructions constitute a processor. Further, the processors have to access some data or signal in order to be put to sleep. It doesn't "just" happen spontaneously.

On page 14, under section *7.4*, the Appellant argues, "Second, even if one accepts the premise that Bowers' controller, as one of the plurality of processors, could also be put to sleep for replacement as directed by one of the other processors in Bowers' system. This type of operation, claimed by the Appellant, is not possible using Bowers' system." The Examiner respectfully disagrees. In any of claims 5-16 and 24-

26, the Appellant never claims that the monarch processor could be any of the plurality of processors. Instead, it appears that there is always one processor pre-selected to be the monarch processor, and that the monarch processor is not capable of being any other processor. For this reason, it is not necessary for the controller of Bowers to be able to be put to sleep.

On page 15, under section *7.4*, the Appellant argues, "Therefore, Bowers does not teach or suggest a 'monarch processor being capable of executing the error handling routine to correct the error...' as claimed by the Appellants in independent claim 5 (and dependent claims 6-7) such that 'the monarch processor is capable of sending a wake up signal to the plurality of slave processors to exit the rendezvous state' as claimed in claim 7." The Examiner respectfully disagrees. In column 5, lines 2-3, Bowers discloses a controller (monarch processor included in the computer system) that generates a stop clock signal STPCLK# and a sleep signal SLP# (error handling routine to correct an error). And in column 7, lines 4-6, Bowers discloses that when all removals or replacements have been made, the controller will begin the reinitialization sequence to return the computer to normal operation (wherein the monarch processor is capable of sending a wake up signal to the plurality of slave processors to exit the rendezvous state).

On page 15, under section *7.4*, the Appellant argues, "Further, Bowers does not teach or suggest 'a plurality of processors including a monarch processor ... and an interrupt signaling mechanism ... to initiate a rendezvous state ... being a state where all of the plurality of processors but the monarch processor are idle' as claimed by the

Appellants in independent claim 8 (and dependent claims 9-11). In addition, Bowers does not teach or suggest 'a plurality of processors ... and an operating system layer ... to signal all but one of the plurality of processors to end a rendezvous state ... upon receiving a signal that error handling is completed, said rendezvous state being a state wherein all but the one of said plurality of processors are idle' as claimed by the Appellants in independent claim 12 (and dependent claims 13-15)." The Examiner respectfully disagrees for at least the reasons given in the rejection above. Further, the Appellant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

On page 15, under section *7.4*, the Appellant argues, "Bowers also does not teach or suggest 'detecting an error ...; entering a rendezvous state in which all processors but the one processor included in the multiple processor system are idle; ... and ... correcting the error using the one processor' as claimed by the Appellants in independent claim 15 (and dependent claims 16-17). Finally, Bowers does not teach or suggest 'attempting to correct an error ... in a multiple processor system ... and on failure, entering a rendezvous state to correct the error, said rendezvous state being a state where all but one of the processors included in the multiple processor system are idle' as claimed by the Appellants in claim 24 (and dependent claims 25-26)." The Examiner respectfully disagrees for at least the reasons given in the rejection above. Further, the Appellant's arguments fail to comply with 37 CFR 1.111(b) because they

amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

On pages 15-16, under section *7.4*, the Appellant argues, "Falik suffers from similar deficiencies. Specifically, Falik fails to disclose 'attempting to correct an error by a detecting processor included in a multiple processor system' and 'entering a rendezvous state to correct the error, said rendezvous state being a state where all but one of the processors included in the multiple processor system are idle' as claimed in claims 18 and 19. While the Office Action asserts that Falik's debugger interface is somehow equivalent to a 'detecting processor included in a multiple processor system', this does not comport with the clear distinction Falik makes between the host computer 1820 and the multiprocessor integrated circuit 1810." The Examiner respectfully disagrees for at least the reasons given in the rejection above. The Examiner is unsure as to what the Appellant means by "this does not comport with the clear distinction Falik makes between the host computer 1820 and the multiprocessor integrated circuit 1810." The Examiner has shown in the rejection what elements of Falik are equivalent to the Appellant's claimed elements and is unsure as to what this distinction has to do with anything.

On page 16, under section *7.4*, the Appellant argues, "Even if it is assumed that Falik's debugger can operate as a 'detecting processor' how does Falik's system correct the error? Falik's debugger, or a monitor, are the only resources available, and neither one operates to 'correct' the error. It is respectfully noted that the term "error" appears

only once in Falik, concerning bus communication error probability. However, such errors are not processed by Falik's system. See Falik, Col. 18, lines 31-32. In fact, Falik states that, while such errors can be cured by a system reset, such operation should be 'avoided by the host' since this resets the entire chip. See Falik, Col, 18, lines 39-47." The Examiner respectfully disagrees. It is noted that Falik is concerned with a debugger. To debug by definition is "to detect, locate, and **correct** logical or syntactical **errors** in a program or **malfunction** in hardware."[1] In column 2, lines 43-46, Falik et al. disclose that the debugging phase completed, therefore, the error was corrected.

On page 16, under section *7.4*, the Appellant argues, "The Appellant agrees that the term debug, by definition, may include correcting logical or syntactical errors. However, there [sic] mere fact that debugging activity occurs does not mandate any particular agency by which the actions are accomplished. For example, debugging typically occurs using human decision processes to propose and test solutions. This not the same as what is claimed by the Appellant, where the a [sic] detecting processor itself can operate by 'attempting to correct an error...'. This type of activity is not disclosed by Falik." The Examiner respectfully disagrees. The Appellant's assertion that a human is involved in the debugging process of Falik is not supported by Falik. In column 2, lines 39-40 and lines 48-49, Falik discloses a host computer having a debugger and the debugger software executing on the processors. The debugger appears to be a software program that is capable of correcting errors. There isn't any mention of a human providing the debugging.

---

[1] Microsoft Computer Dictionary, Fifth Edition, 2002, page 148.

On page 16, under section *7.4*, the Appellant argues, "Finally, how can 'all but one of the processors included in the multiple processor system' be idle, as asserted in the Office Action, if at least one processor in the multiprocessor integrated circuit 1810 must be awake to execute a monitor, *in addition* (emphasis by Appellant) to the debugger of Falik's host computer? The debugger communicates with the monitor on one of the processors, which means at least two processors must be operational in Falik's system. See Falik, Col. 17, lines 27-46." The Examiner respectfully disagrees. The Examiner has examined Falik, col. 17, lines 27-46 and is unsure as to where the Appellant gets the notion that at least two processors must be operational. It is clear that all of the monitors on the processors are brought back up and synchronized after the error is corrected. Further, in column 7, lines 26-30, Falik et al. disclose that the interrupt control module issues an ISE interrupt request to either a specific one of the processors or to multiple processors (on failure, entering a rendezvous state to correct the error, said rendezvous state being a state where all but one of the processors included in the multiple processor system are idle).

On page 17, under section*7.5*, the Appellant argues, "First there is no motivation to combine Bowers and Fujii. Bowers never uses the term 'error', or discloses any type of error detection or handling routine. Processor boards are simply replaced after the fact – notably, for routine maintenance. Similarly, Fujii never mentions removing a processor from a system." The Examiner respectfully disagrees. The Examiner would like to note once again that the number of times the word error appears in a reference is an ineffective way to show that errors did or didn't occur. To replace a processor for

routine maintenance indicates that there is something wrong with the processor and that it was causing errors as shown in Bowers: column 1, lines 66-67 continued in column 2, lines 1-11. In response to appellant's argument that there is no suggestion to combine the references, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988)and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992). In this case, an error event type is used to report a non-recoverable problem (see Fujii et al.: col.2, lines 62-63) and a warning event type is used to indicate some kind of recoverable anomaly (see Fujii et al.: col. 2, lines 60-62). Knowing the severity determines what action should be taken (removal of a processor) (see Fujii et al.: col. 2, lines 48-55).

On page 17, under section 7.5, the Appellant argues, "Second the Office mischaracterizes the claimed 'severe error' (which causes entry into a rendezvous state) as a 'non-recoverable problem' documented by Fujii. However, this characterization does not comport with the concept of a 'rendezvous state' claimed by the Appellants." The Examiner respectfully disagrees. It is noted that the features upon which appellant relies (i.e., Application page 7, lines 19-24) are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). Claim 17 fails to mention that the error is

recoverable or non-recoverable, but rather, refers to the error as being severe. A non-recoverable error is severe. For this reason it is reasonable for the Examiner to interpret it as such.

On pages 17-18, under section *7.5*, the Appellant argues, "Third, it is not necessarily true that 'knowing the severity of the event determines what action should be taken (e.g., removal of a processor)' as asserted in the Office Action. Errors of varying severity can occur within a computer system, and knowledge of the severity (e.g., non-recoverable) will not always be determinative as to whether a processor should be replaced (e.g. a main memory failure may be the culprit, or a hard disk failure, etc.). Since there is no evidence in the record to support the Office Action assertion, the explicit requirements set forth by *In re Sang Su Lee* are not satisfied." The Examiner respectfully disagrees. The primary reference of Bowers is concerned with replacing processors and only that. Therefore, any error detected in Bowers is concerned with just the processors. Bowers is silent as to how it is determined if the processor should be replaced. Fujii cures this deficiency by relating the severity of errors with appropriate actions. The Examiner doesn't rely on personal knowledge, but rather what is taught by each reference.

On page 18, under section *7.5*, the Appellant argues, "Fourth, it should be noted that neither Bowers nor Fujii disclose using a processor that is part of a multi-processor system to correct an error within the system, as claimed by the Appellant." The Examiner respectfully disagrees for at least the reasons given in the rejection above.

On page 18, under section *7.5*, the Appellant argues, "Fifth, combining Bowers with Fujii gives no reasonable expectation of success. Fujii merely teaches the existence of an error recording system, not a system to correct errors. The mechanism of replacing a processor (as taught by Bowers) in response to recording non-recoverable errors, advocated in the Office Action, also may not have any effect on solving the actual problem (e.g., a main memory failure, or a cache failure)." The Examiner respectfully disagrees for at least the reasons given in the argument above.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

ROBERT W. BEAUSOLIEL
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

MM

Conferees:

Robert Beausoliel

Lynne Browne

LYNNE H. BROWNE
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100